Prototype 1 and Customer Feedback

Shehryar Ali Memon

Oluwatamilore Ilupeju

Logan Jones

Saheel Ahmed

March 05, 2022

Abstract

This document is meant to portray our initial prototypes and analyze our inverse kinematics solutions for our end-effector. It also serves the purpose of defining any challenges we faced during prototyping and makes clear our plan for our next prototype.

Table of Contents

Introduction

This document follows up on the last document, in which we defined our prototype test plan and our stopping criteria. This document contains images of the initial prototypes of our sander, inverse kinematics solution, and Python code for our UI. It further defines our next steps for our future prototypes by virtue of another prototype test plan and makes known the software and hardware challenges we faced during prototyping. An updated bill of materials (BOM) can also be found attached to this document, in which our expenses have been minimized.

Prototyping test plan

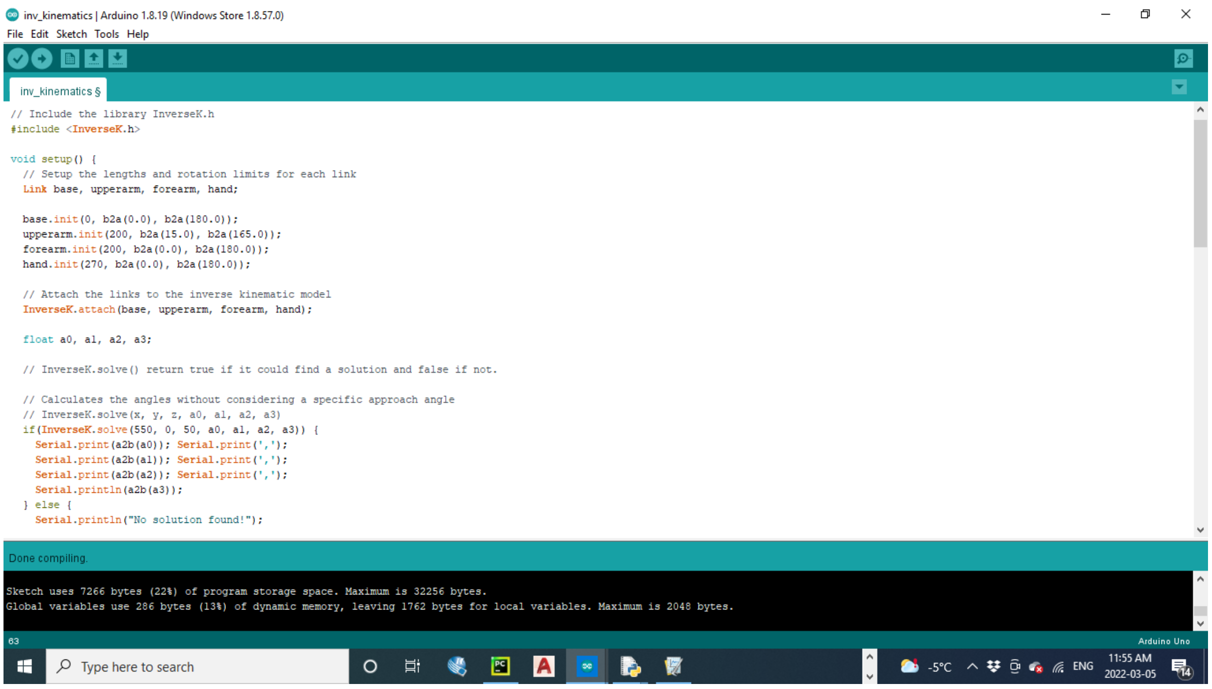
Test ID	**Test Objectives**	**Description of prototypes and basic test method**	**Description of results to be recorded and how these results will be used**	**Estimated test duration and planned start date**
01	Power on/off	Checking whether system starts up and shuts down properly	N/A	03/10/2022 15 minutes
02	Primary axes Locomotion	Accuracy of movement robot on all axes using inverse kinematics	If it fails to move accordingly, the inverse kinematics solution will be revisited	03/10/2022 2 hours
03	Input Latency	Delay between movement input and output	Time taken to satisfy a condition. Goal is to reduce time as much as possible.	03/10/2022 2 hours

04	Ease of attachment	Swapping in/out all attachments	Time taken to change an attachment; latch system will be revisited.	03/10/2022 1 hour(s)
05	Paint gun 3D render	Final model of paint gun	Once approved, model will be 3D printed	03/09/2022-03/11/2022 3 days
06	Sander 3D print	Final product of sander	Efficiency checked once printed	03/13/2022 10 hours
07	Vacuum Efficiency	Amount of residue picked up	Difference in mass in grams of vacuum bag per hour	03/13/2022 2 hours
08	Sander Efficiency	Area cleared in a period of time along with accuracy and precision	Area in m^2 cleared per hour, difference checked between the desired area and cleared area	03/13/2022 2 hours
09	Usability	Overall ease of use ranked by users	Comfortability with the system; ergonomics and design rechecked	03/13/2022 1 hour(s)
10	Safety	Overall safety of system checked by obstruction testing and coding IR sensor	Safety and viability checked; Arduino solution will be revisited	03/13/2022 1 hour(s)

Prototype Analysis

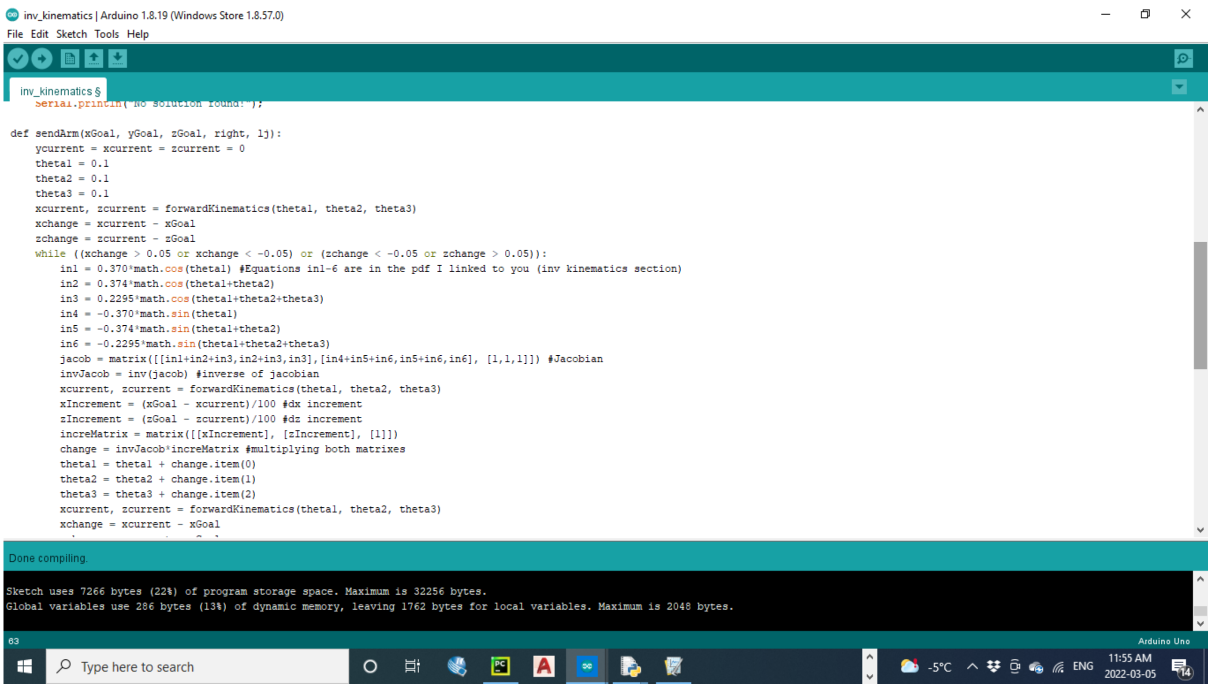
Inverse kinematics solver (low fidelity)

Provided below is the prototype of the inverse kinematics solver for the 3-DOF robot arm:



```
// Include the library InverseK.h
#include <InverseK.h>

void setup() {
  // Setup the lengths and rotation limits for each link
  Link base, upperarm, forearm, hand;

  base.init(0, b2a(0.0), b2a(180.0));
  upperarm.init(200, b2a(15.0), b2a(165.0));
  forearm.init(200, b2a(0.0), b2a(180.0));
  hand.init(270, b2a(0.0), b2a(180.0));

  // Attach the links to the inverse kinematic model
  InverseK.attach(base, upperarm, forearm, hand);

  float a0, a1, a2, a3;

  // InverseK.solve() return true if it could find a solution and false if not.

  // Calculates the angles without considering a specific approach angle
  // InverseK.solve(x, y, z, a0, a1, a2, a3)
  if(InverseK.solve(550, 0, 50, a0, a1, a2, a3)) {
    Serial.print(a2b(a0)); Serial.print(',');
    Serial.print(a2b(a1)); Serial.print(',');
    Serial.print(a2b(a2)); Serial.print(',');
    Serial.println(a2b(a3));
  } else {
    Serial.println("No solution found!");
```

```
Sketch uses 7266 bytes (22%) of program storage space. Maximum is 32256 bytes.
Global variables use 286 bytes (13%) of dynamic memory, leaving 1762 bytes for local variables. Maximum is 2048 bytes.
```

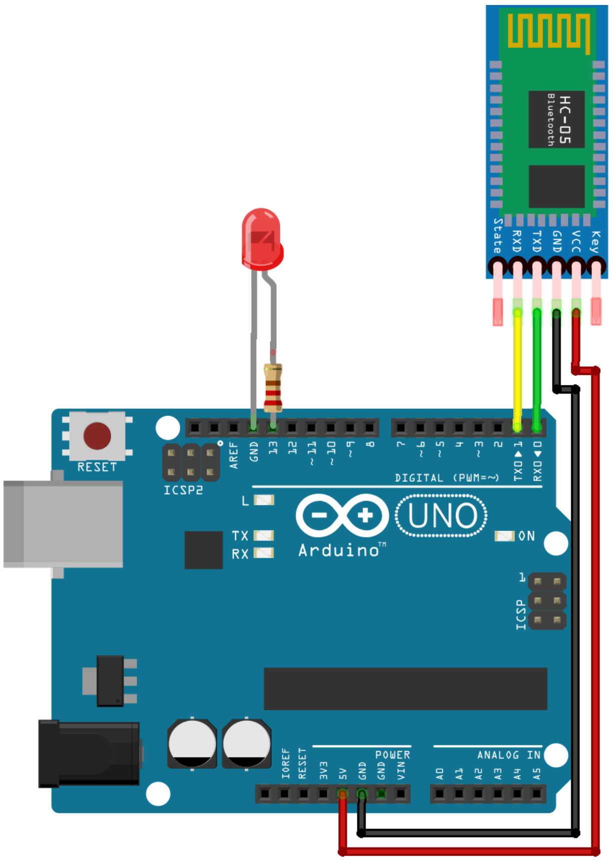


```
    Serial.println("No solution found!");

def sendArm(xGoal, yGoal, zGoal, right, lj):
    ycurrent = xcurrent = zcurrent = 0
    theta1 = 0.1
    theta2 = 0.1
    theta3 = 0.1
    xcurrent, zcurrent = forwardKinematics(theta1, theta2, theta3)
    xchange = xcurrent - xGoal
    zchange = zcurrent - zGoal
    while ((xchange > 0.05 or xchange < -0.05) or (zchange < -0.05 or zchange > 0.05)):
        in1 = 0.370*math.cos(theta1) #Equations in1-6 are in the pdf I linked to you (inv kinematics section)
        in2 = 0.374*math.cos(theta1+theta2)
        in3 = 0.2295*math.cos(theta1+theta2+theta3)
        in4 = -0.370*math.sin(theta1)
        in5 = -0.374*math.sin(theta1+theta2)
        in6 = -0.2295*math.sin(theta1+theta2+theta3)
        jacob = matrix([[in1+in2+in3,in2+in3,in3],[in4+in5+in6,in5+in6,in6], [1,1,1]]) #Jacobian
        invJacob = inv(jacob) #inverse of jacobian
        xcurrent, zcurrent = forwardKinematics(theta1, theta2, theta3)
        xIncrement = (xGoal - xcurrent)/100 #dx increment
        zIncrement = (zGoal - zcurrent)/100 #dz increment
        increMatrix = matrix([[xIncrement], [zIncrement], [1]])
        change = invJacob*increMatrix #multiplying both matrixes
        theta1 = theta1 + change.item(0)
        theta2 = theta2 + change.item(1)
        theta3 = theta3 + change.item(2)
        xcurrent, zcurrent = forwardKinematics(theta1, theta2, theta3)
        xchange = xcurrent - xGoal
```

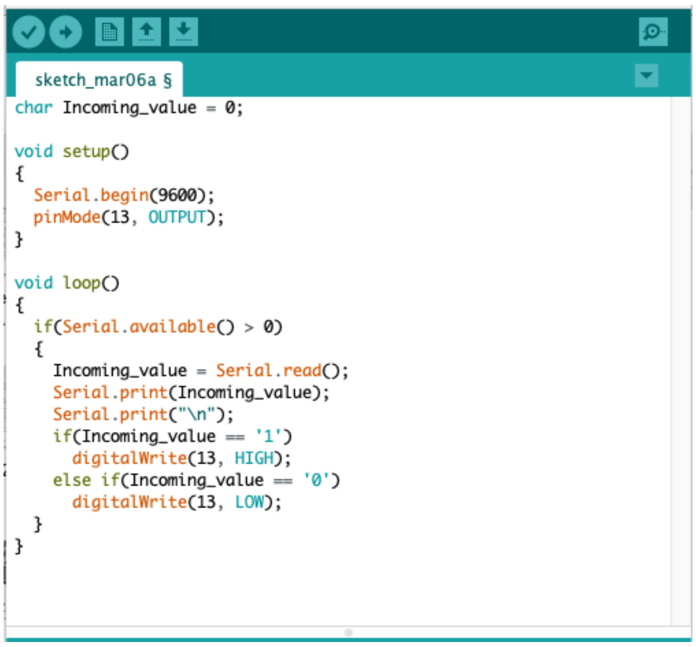
```
Sketch uses 7266 bytes (22%) of program storage space. Maximum is 32256 bytes.
Global variables use 286 bytes (13%) of dynamic memory, leaving 1762 bytes for local variables. Maximum is 2048 bytes.
```

Graphical User Interface

Bluetooth



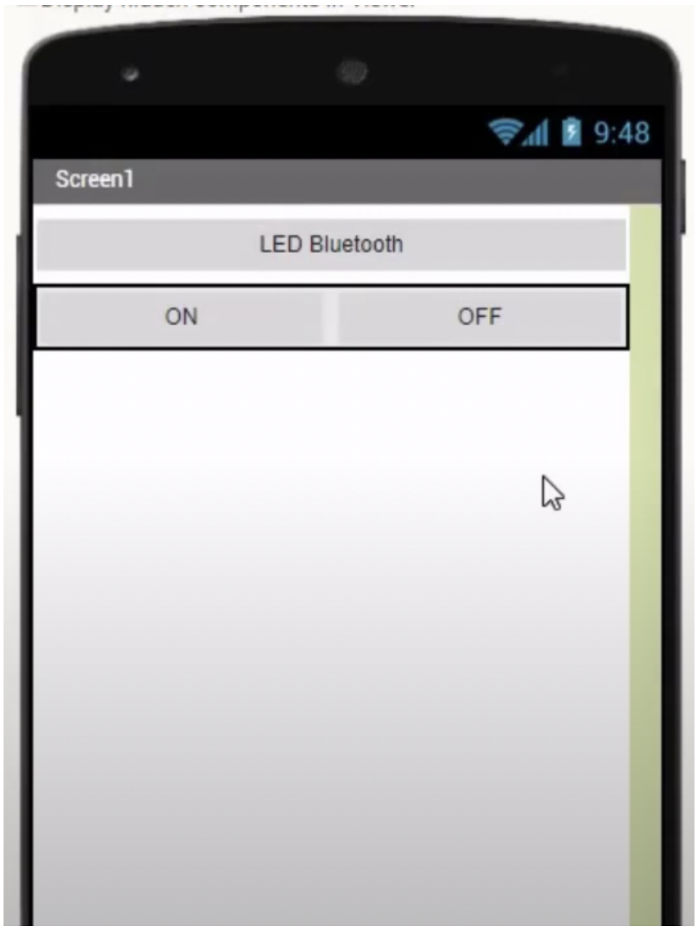
HC-D5 bluetooth module

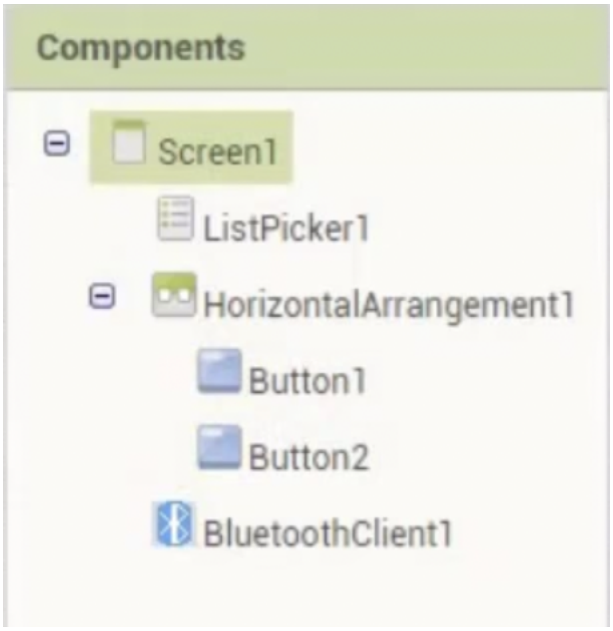


```
char Incoming_value = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void loop()
{
  if(Serial.available() > 0)
  {
    Incoming_value = Serial.read();
    Serial.print(Incoming_value);
    Serial.print("\n");
    if(Incoming_value == '1')
      digitalWrite(13, HIGH);
    else if(Incoming_value == '0')
      digitalWrite(13, LOW);
  }
}
```

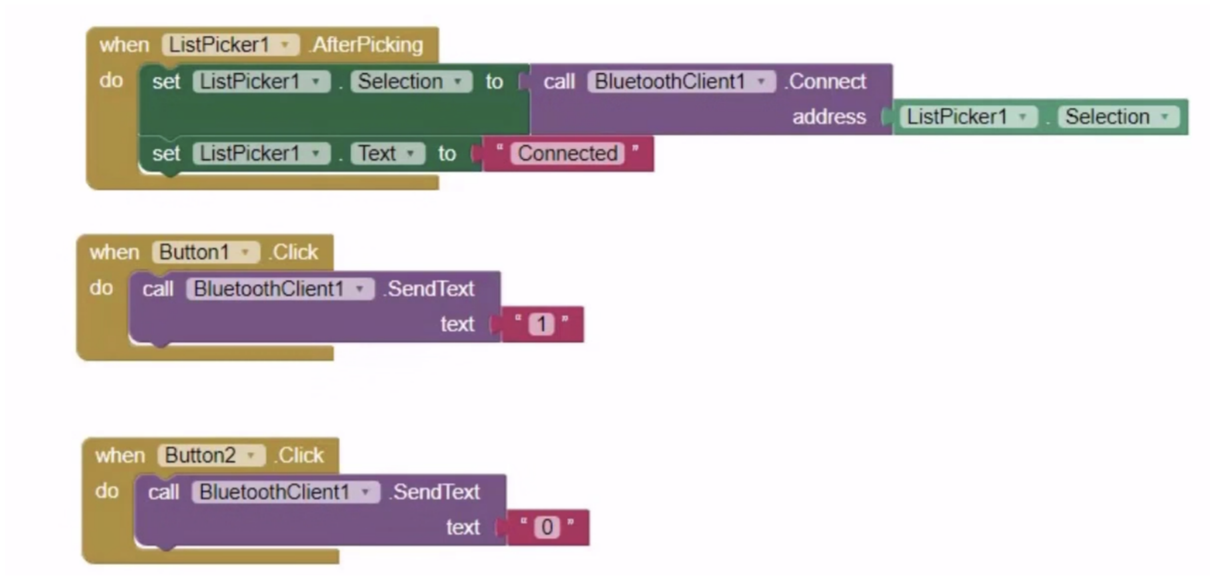
Arduino code that allows Bluetooth connectivity

Application





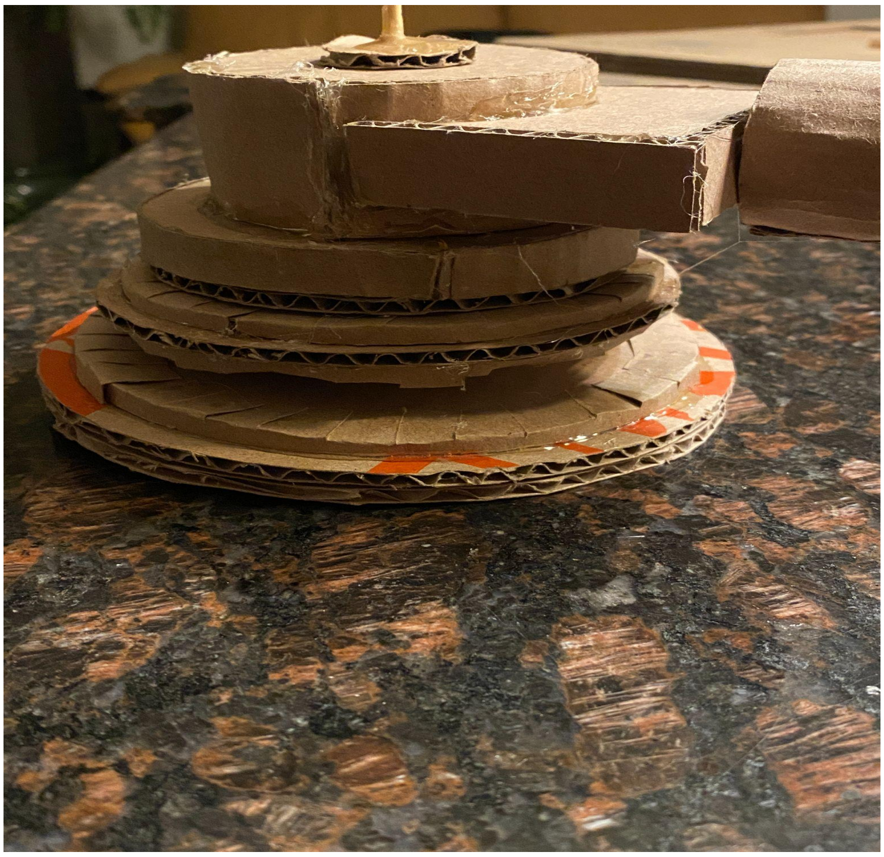
App Components through MIT App Inventor



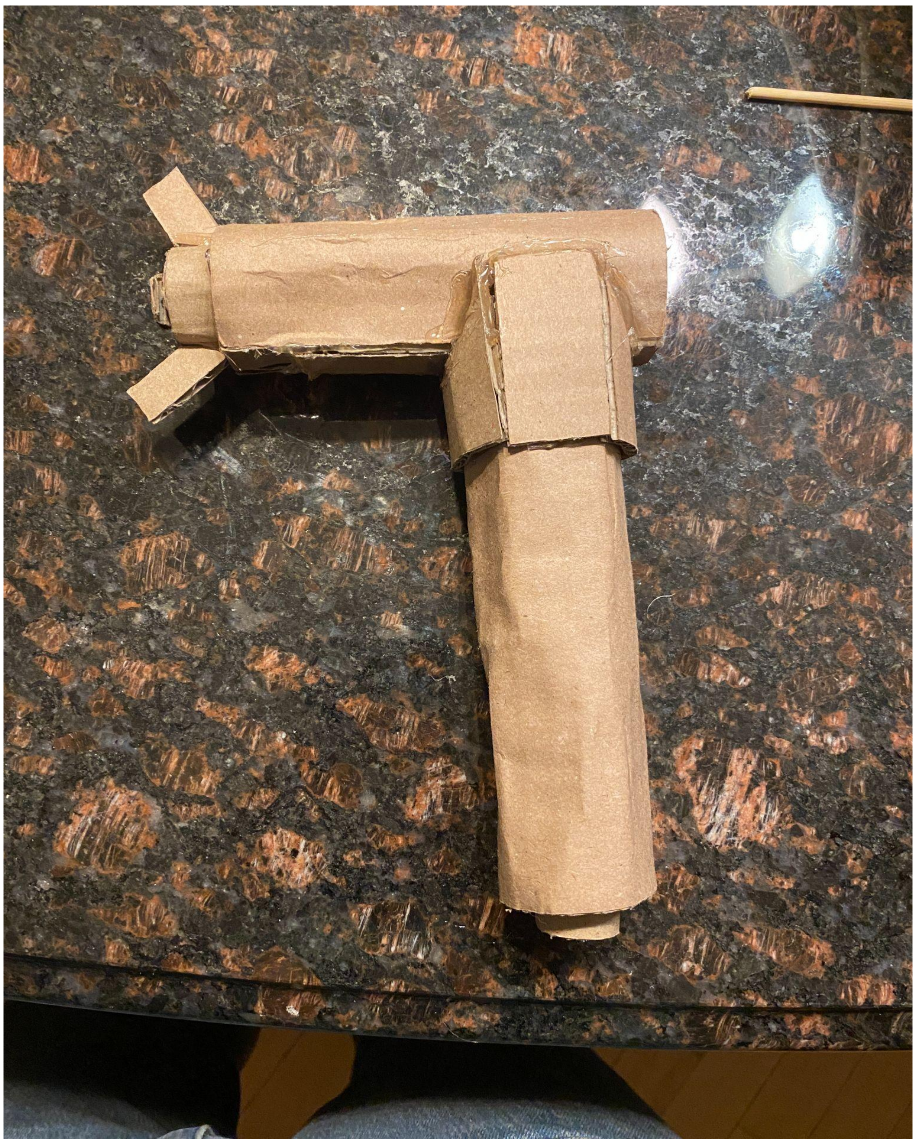
```
when ListPicker1 .AfterPicking
do  set ListPicker1 . Selection to  call BluetoothClient1 .Connect
                                                  address  ListPicker1 . Selection
    set ListPicker1 . Text to " Connected "

when Button1 .Click
do  call BluetoothClient1 .SendText
                            text " 1 "

when Button2 .Click
do  call BluetoothClient1 .SendText
                            text " 0 "
```

App Components through MIT App Inventor (eg. LED)

Sander Prototype



Paint Gun Prototype



In order to save funds for the final product, these prototypes are constructed out of household materials, mainly corrugated cardboard and hot glue. These first two prototypes are just a basic proof of concept, to see how large the effectors would be in real life and that the mechanical parts could fit in the housing. Find attached the link to the working of our Sander - Spinning Sanding Pad.

Feedback

- Client wanted to see proper safety precautions to mitigate any failsafe issues, so we included an IR sensor in our designs
    - The IR sensor would be placed at the base of the arm, facing upwards at a 45-degree angle
    - Arduino program for the sensor nearing completion

- IK solution has become our main priority
    - Solution is still difficult to come by
    - Basic understanding of the problem has been built

- Remote Application is in development for an inviting GUI experience
    - Alpha model is near completion

- Workable prototype of the sander created

- Camera idea has been dropped for this prototype
    - Will be revisited in the latter stages of the project

Challenges

Inverse Kinematics Solver

- Understanding and implementing inverse kinematics code in Python
- Converting Python code into Arduino IDE code
- Reducing the length of the code
- Understanding all appropriate variables/inputs

Graphical User Interface

- Understanding how to use bluetooth to control the arduino uno
- Understanding the use of MIT App inventor to show video output
- Making the app easy to use
- Making the app easy to update / change
- Understanding how to connect the arduino to the phone

Sander Prototype

- Finding an appropriate adhesive to hold the parts of the sander together. That won't fail due to the vibrations of the motor
- Hooking the sander up to a power source without interfering with the movement of the arm

Paint Gun Prototype

- Finding an appropriate air compressor to force the paint out of the gun

Wrike Updates:
https://www.wrike.com/frontend/ganttchart/index.html?snapshotId=j6CtV1ikmcJ4Z2GzUi5TnVds3sikcHVu%7CIE2DSNZVHA2DELSTGIYA